

FIGURE 1

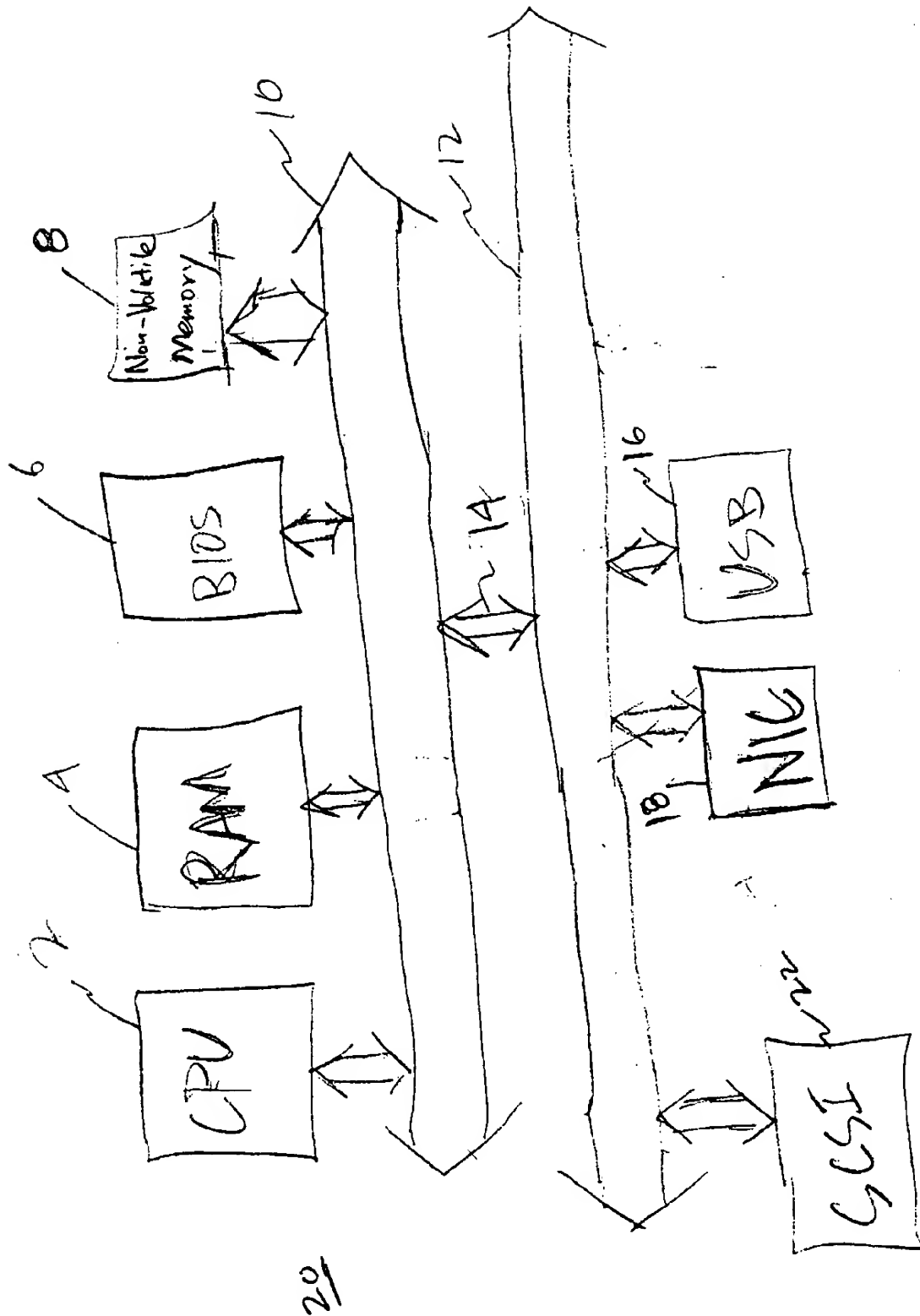


Figure 1

FIG. 2

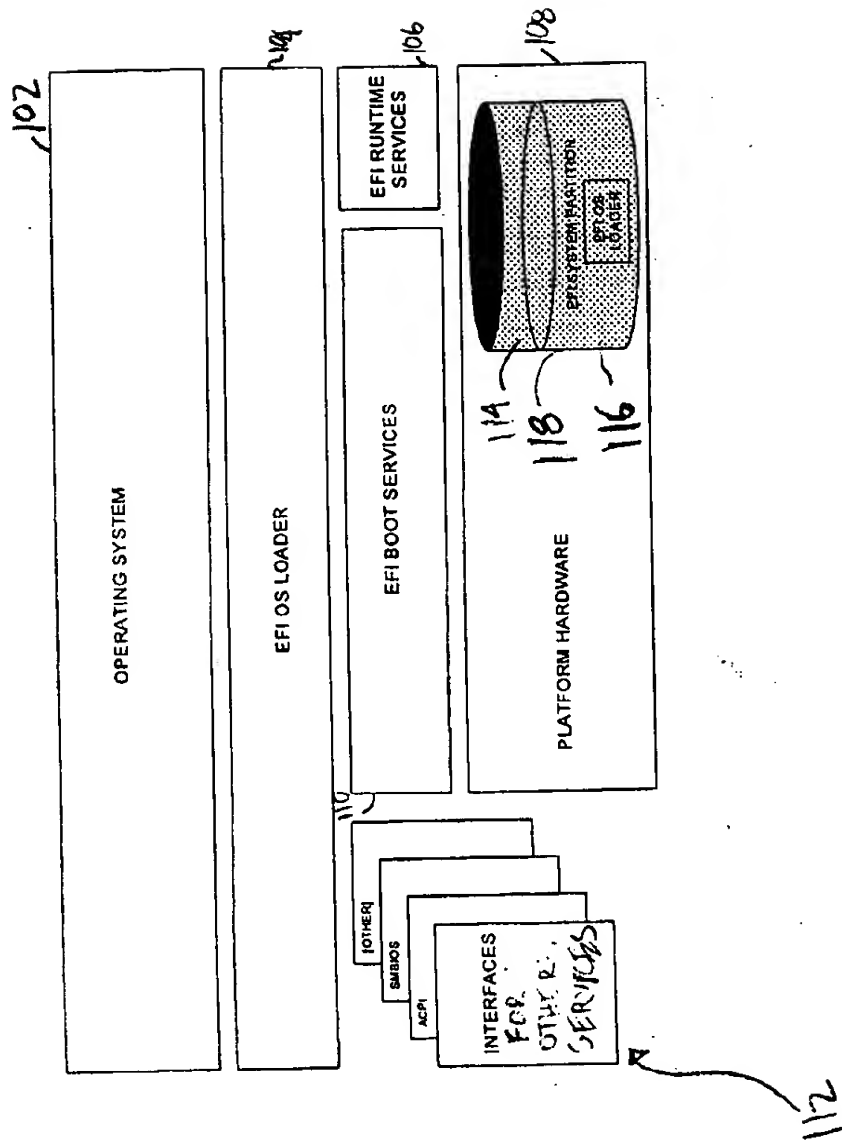


Figure 2

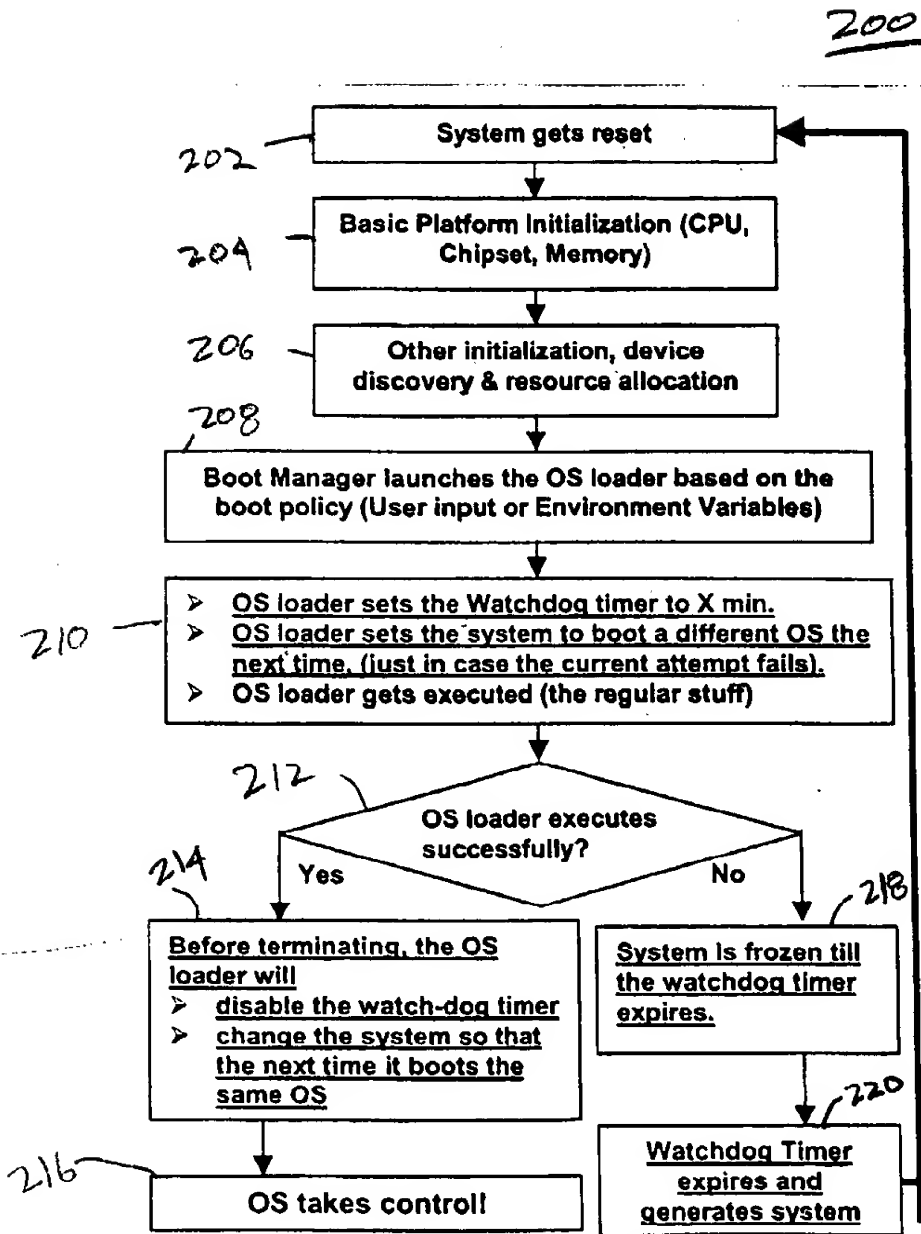


Figure 3

300

// Each OS loader can choose their own timer duration dependent on how long they need to boot  
 SetWatchDogTimer(360, //set the watchdog timer for 6min - 5 \* 60 seconds  
 0xFFFF, //OEMs can choose any number other than those between 0x0000-0xFFFF, 0);

// Set the BootNext variable to the appropriate value. It will be used in the next reboot if the current attempt fails.  
 // BootCurrent will have the OS loader that will be used for the current boot. This will be set by the boot manager.  
 // So get that value in the variable "CurrentlySelectedBootLoader"  
 GetVariable ( L"BootCurrent"....., CurrentlySelectedBootLoader);  
 // We have to find out what OS loader is next to the currently selected one in the system's BootOrder.  
 GetVariable (L"BootOrder"....., SystemBootOrder);  
 // Get the number of elements in SystemBootOrder. BootOrder is an array of UINT16s.  
 int LengthOfArray = sizeof (SystemBootOrder) / sizeof (UINT16);  
 // Now scan through the list of boot options in Boot Order. (BootOrder is an ordered list of boot options)  
 for (int i = 0; i < LengthOfArray; i++)  
 {  
 if (i == LengthOfArray - 1) //Extreme condition when no other boot options are available to try.  
 {  
 Exit(); //Exits to the firmware shell it has exhausted all the possible OS boot options.  
 // This behavior can be customized by the OEMs to alert an administrator or do other things.  
 }  
 // if the given element in the array is the currently selected OS loader stored in BootCurrent, then...  
 if (SystemBootOrder[i] == CurrentlySelectedBootLoader)  
 {  
 // If the current boot attempt fails, we must boot the next OS as specified in the BootOrder  
 // So, set the BootNext variable to the appropriate value i.e., the next element in the BootOrder  
 // list  
 SetVariable(L"BootNext",.....sizeof(UINT16), SystemBootOrder[i+1]);  
 break;  
 }  
 }  
}

// OS loader does its regular functions - initializations, loading the drivers, OS kernel, etc.

OS loader boots the OS successfully?  
 (OS loader terminates with no errors)

Yes

No

// Before terminating, the OS loader will do the following:  
 // Disable the watchdog timer, otherwise the system might get  
 // reset when the timer expires.  
 SetWatchDogTimer (0, //A value of zero disables the watchdog  
 .....);  
 //Delete the BootNext variable so that the next time  
 //the system does not boot a different OS.  
 SetVariable ( L"BootNext", .....  
 0, //Data size is set to 0 which deletes the variable  
 NULL); //pointer to data is NULL!  
 //All set for the Operating System to take over. So, the firmware  
 //can relinquish its control by calling ExitBootServices  
 ExitBootServices (.....);

OS takes control!

Wait for the watchdog timer to expire.  
 (OS loader failed to boot the OS and the  
 system has frozen. Just wait for the  
 watchdog to generate a reset!)

Watchdog timer expires and the system  
 gets reset.

Figure 4